Iterative Coded Pulse-Position-Modulation for Deep-Space Optical Communications

Maged F. Barsoum
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, CA 91109
mbarsoum@jpl.nasa.gov

Bruce Moision
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, CA 91109
bmoision@jpl.nasa.gov

Michael Fitz
Department of Electrical Engineering
UCLA
Los Angeles, CA 90095
fitz@ee.ucla.edu

Dariush Divsalar
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, CA 91109
Dariush.Divsalar@jpl.nasa.gov

Jon Hamkins
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, CA 91109
jon.hamkins@jpl.nasa.gov

***Abstract*— This paper presents and compares two iterative coded modulation techniques for deep-space optical communications using pulse-position modulation (PPM). The first code, denoted by SCPPM, consists of the serial concatenation of an outer convolutional code, an interleaver, a bit accumulator, and PPM. The second code, denoted by LDPC-PPM, consists of the serial concatenation of an LDPC code and PPM. We employ Extrinsic Information Transfer (EXIT) charts for their analysis and design. Under conditions typical of a communications link from Mars to Earth, SCPPM is 1 dB away from capacity, while LDPC-PPM is 1.4 dB away from capacity, at a Bit Error Rate (BER) of approximately 10^{-5}. However, LDPC-PPM lends itself naturally to low latency parallel processing in contrast to SCPPM.**

I. Introduction

A deep-space communications link must have a transmitter that is both power efficient and low weight. Optical links satisfy these requirements, and also can use large bandwidths, making them a viable candidate to replace or supplement current deep-space radio-frequency links. In this paper, we present efficient error-control-coding schemes for a deep-space optical link, where peak and average power constraints dictate a small duty cycle. These codes were developed to address the needs of a communications link from Mars to Earth, designed to transmit at mega-bit per second rates.

A direct-detection optical link may be modeled as a Poisson point process, with system constraints imposing peak power, average power, and bandwidth limitations. Wyner [1] showed that under peak and average power constraints, negligible capacity loss is incurred when restricting the modulation to a binary, slotted scheme. Shamai [2] extended this result to include a bandwidth constraint, illustrating regions where a binary, slotted scheme is near-optimal. For peak power, average power, and bandwidth constraints typical of a deep-space link, restricting the modulation to pulse-position-modulation (PPM) is near-capacity achieving [3]. A number of authors noted the efficiency of PPM for an optical channel prior to these results, e.g., [4].

Hence, we may, with negligible loss relative to alternatives, fix the modulation to be PPM, and address the choice of a suitable Error Control Code (ECC). A number of authors have addressed the design of ECC's for the Poisson PPM channel. In an earlier work [5], a Reed-Solomon (RS) code was proposed for error protection on this channel. A noiseless (meaning no background photons) Poisson PPM channel reduces to a symbol erasure channel, and an (n, k) RS code may be tailored to fit an M-ary PPM channel by choosing $n = M - 1$ and taking code symbols from the Galois field with M elements.

However, the deep-space optical channel is seldom noiseless. Moreover, a block length of $n = M - 1$ is, for cases of practical interest, too small to achieve good performance. Longer block lengths may be obtained with RS codes defined in higher-order fields, in which multiple PPM symbols are associated with each code symbol, but this results in only marginal improvement [6]. RS performance on a noisy Poisson channel typically remains 3 dB or more away from capacity when conventional hard-decision decoding is used [6].

With the advent of turbo codes, several authors investigated the application of parallel concatenated codes (PCCs) to the PPM channel. A PCC was applied to the binary PPM channel [7], and to the M-ary PPM channel [8]. In the latter paper the PPM demodulator passes soft information to the turbo code but is not involved in iterative decoding. These approaches allowed performance improvements over RS coded PPM but failed to capitalize on iterative demodulation. Iterative demodulation of PPM with a PCC was treated in [9], applied to the discrete-time Rayleigh fading channel, where low duty cycles are also optimum.

In this paper we examine two ECCs designed for the PPM channel. The first is a serially concatenated code, in the sense presented in [10], where the constituent codes are a convolutional code and PPM preceded by a bit-accumulator.

Details of this code are described in [11]. Its combination of a bit-accumulator and PPM is referred to as accumulate-PPM (APPM). The second is also a serially concatenated code, where the constituent codes are a low-density-parity-check (LDPC) code and PPM. In both cases, PPM or APPM are iteratively demodulated. Both of these approaches provide better performance than non-iterative alternatives discussed above, and better complexity and performance than the iterative alternatives.

The remainder of this paper is organized as follows. In Section II we describe the channel model. In Section III we find the Extrinsic Information Transfer (EXIT) functions for PPM and APPM, which are instrumental in code design. In Section IV, we describe the first code that uses APPM. In Section V, we describe the LDPC code. In Section VI, we discuss the complexity and performance of both codes.

Notation is as follows. Lowercase u, w, y, x denote realizations of the corresponding random variables U, W, Y, X. Boldface $\mathbf{u} = (u_1, u_2, \ldots, u_n)$ and $\mathbf{U} = (U_1, \ldots, U_n)$ denote vectors. $\mathbf{w}_{[k]}$ denotes the vector $\mathbf{w}$ with element w_k removed. Where clear or irrelevant the subscript k of an element u_k from vector $\mathbf{u}$ may be dropped for simplicity. The notation $p_Y(y)$ is used to denote the probability density or mass function of random variable Y evaluated at y. When the random variable is clear from the context, we simply write $p(y)$ for $p_Y(y)$.

II. Channel Model

For the purpose of ECC design, the channel model used throughout this paper is defined as follows. Binary symbols $\mathbf{c}$ are transmitted over the optical channel and received as $\mathbf{y}$. The channel $p(y|c)$ is modeled as a binary-input Poisson channel. In any time slot, either a pulse is transmitted to send a 1, or no pulse is transmitted to send a 0. Letting n_s be the mean received signal photons per pulsed slot and n_b the mean received noise photons per slot, we have

$$p_0(y) \triangleq p_{Y|C}(y|0) = \frac{e^{-n_b} {n_b}^y}{y!} \quad (1)$$

$$p_1(y) \triangleq p_{Y|C}(y|1) = \frac{e^{-(n_b+n_s)} (n_b+n_s)^y}{y!} \quad (2)$$

Channel uses are assumed to be conditionally independent, i.e.,

$$p(\mathbf{y}|\mathbf{c}) = \prod_k p(y_k|c_k) \quad (3)$$

and bits are modulated using M-ary PPM. In PPM, each $\log_2 M$ bits map to the location of a single pulsed slot in an M-slot frame. Performance of coded systems will be measured relative to the capacity of a PPM modulated channel. The duty cycle of M-ary PPM is $1/M$. Link budgets for deep-space optical links show that the optimum duty cycle is less than $1/32$ for the entirety of a typical (in orbit) mission [6], [12]. As noted earlier, for these duty cycles, we see a negligible loss in capacity when choosing PPM.

III. PPM and APPM EXIT Curves

In this section we characterize PPM and APPM using extrinsic-information-function-transfer (EXIT) charts. For this, we consider the following scenario. A block of bits $\mathbf{a}$ is encoded by a code $\mathcal{C}$ (either PPM or APPM) to yield the sequence of symbols $\mathbf{c}$. The sequence of symbols $\mathbf{c}$ is transmitted over the channel defined by equations (1-3) and received as $\mathbf{y}$.

The code is decoded via a soft-input-soft-output (SISO) algorithm, or decoder. A SISO decoder receives, as soft inputs, noisy versions, or log-likelihoods, of the input and output of the encoder and produces updated log-likelihoods of the input, or output, or both. These likelihoods may then be transmitted to other SISO modules in the receiver where they are treated as noisy inputs. Derivations of the SISO algorithm have appeared in various forms in the literature, e.g., [13], [14] and will not be presented here.

In our case, the SISO decoder of code $\mathcal{C}$ receives $\mathbf{y}$, a noisy version of $\mathbf{c}$ over the memoryless communications channel $p(y|c)$. We also think of it as receiving a sequence $\mathbf{w}$, a noisy version of $\mathbf{a}$ over a memoryless *extrinsic* channel $p(w|a)$. With the serial concatenation of an outer code with the inner code $\mathcal{C}$, the extrinsic channel models information coming from the outer decoder. The sequence $\mathbf{w}$ and channel $p(w|a)$ are artificial constructs introduced to aid in analysis of the decoder, as is done in [15]. The observations in $\mathbf{w}$ are assumed to be conditionally independent, i.e.,

$$p(w_k, w_j|\mathbf{a}, \mathbf{y}, \mathbf{w}_{[k],[j]}) = p(w_k|a_k)p(w_j|a_j)$$

Information from the noisy observations $\mathbf{w}$ are received by the SISO decoder as a priori log-likelihood ratios (LLRs)

$$l_k = \log \frac{P(A_k = 0|w_k)}{P(A_k = 1|w_k)}$$

from which the SISO algorithm computes, as a function of the a priori LLRs and the mapping of $\mathbf{a}$ to $\mathbf{c}$, extrinsic log-likelihoods

$$l_{e_k} = \log \frac{P\left(A_k = 0|\mathbf{y}, \mathbf{w}_{[k]}\right)}{P\left(A_k = 1|\mathbf{y}, \mathbf{w}_{[k]}\right)}$$

The LLRs

$$l_{p_k} = \log \frac{P(A_k = 0|\mathbf{y}, \mathbf{w})}{P(A_k = 1|\mathbf{y}, \mathbf{w})}$$

are referred to in the literature as the a posteriori information and are the basis for bit decisions. If $p(w|a)$ is an output symmetric channel, meaning that

$$p(w|A = 1) = p(-w|A = 0)$$

then as shown in [16] the conditional densities of the a priori LLRs are symmetric, ie., they satisfy

$$p_{L|A}(-l|1) = p_{L|A}(l|1)\, e^l \quad (4)$$

$$p_{L|A}(-l|0) = p_{L|A}(l|0)\, e^{-l} \quad (5)$$

Empirical evidence shows that $p_{L|A}$ approaches a Gaussian

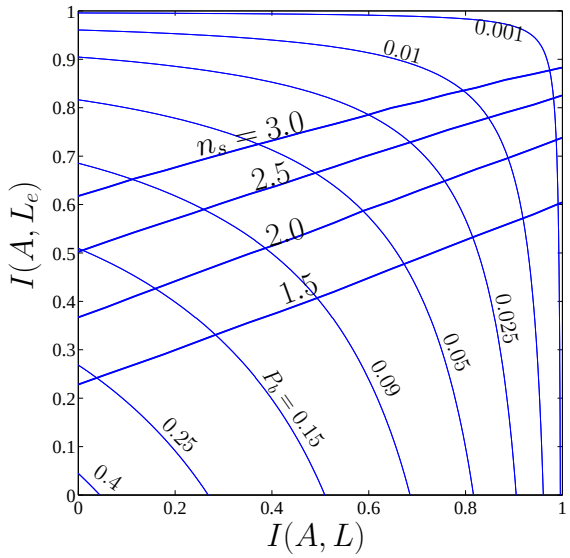

Fig. 1. Output Mutual Information vs Input Mutual Information for the PPM SISO, $n_b = 0.2$, M=64

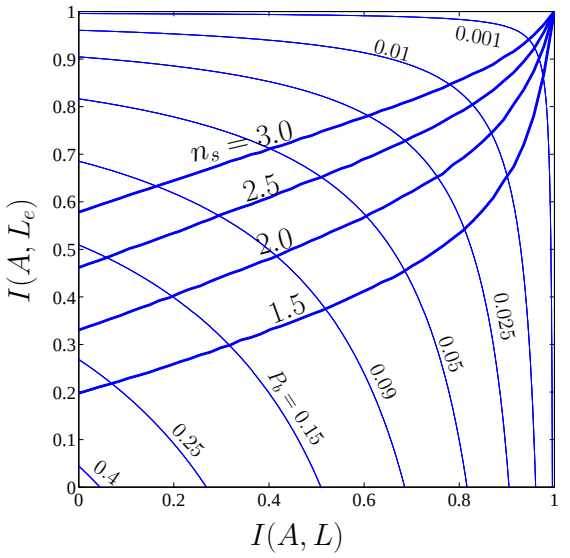

Fig. 2. Output Mutual Information vs Input Mutual Information for the APPM SISO, $n_b = 0.2$, M=64

distribution as the number of iterations grows. This can be intuitively seen as a central limit theorem result. If the conditional LLRs are Gaussian and the channel is output symmetric, equations (4) and (5) imply that

$$E\left[L|A=1\right] = -Var\left[L|A=1\right]/2$$
$$E\left[L|A=0\right] = Var\left[L|A=0\right]/2$$

allowing a simple, single parameter characterization of the LLR statistics.

Ten Brink [17], [18] proposed tracking the evolution of the mutual information between bits and their corresponding LLRs in order to predict the decoder behavior, which has proven to be a useful tool in code design. In our example, an extrinsic information transfer (EXIT) function is a plot of the mutual information $I(A; L_e)$ as a function of $I(A; L)$. For equiprobable binary A, the mutual information $I(A; L_e)$ may be expanded as

$$I(A; L_e) = \frac{1}{2} \sum_{a \in \{0,1\}} \int p_{L_e|A}(l|a) \log_2 \frac{2p_{L_e|A}(l|a)}{p_{L_e|A}(l|0) + p_{L_e|A}(l|1)} dl \tag{6}$$

which may be evaluated via numerical integration using estimates of the densities $p_{L_e|A}$ obtained by simulation. To estimate $p_{L_e|A}$, we assume $p_{L|A}$ is Gaussian, and determine $p_{L_e|A}$ from samples of L_e generated by simulation, and assumed independent and identically distributed. A kernel density estimator along with trapezoidal numerical integration is used, treating the densities as continuous.

Figures 1 and 2 show $I(A; L_e)$ versus $I(A; L)$ for PPM and APPM respectively with $M = 64$, $n_b = 0.2$, and $n_s \in \{1.5, 2.0, 2.5, 3.0\}$. Assuming a symmetric Gaussian distribution for L and L_e, bit error rate contours are also shown from which the error rate in estimating $\mathbf{a}$ can be predicted at any combination of $I(A; L)$ and $I(A; L_e)$.

We note that APPM, which is constructed by preceding PPM with a recursive $1/(1 + D)$ binary accumulator, allows $I(A, L_e)$ to go to 1 as $I(A; L)$ approaches 1, whereas for PPM $I(A, L_e)$ remains below 1 at $I(A, L) = 1$. This comes at the cost of a slightly lower initial $I(A, L_e)$ for APPM compared to PPM, as can be noted by comparing Fig. 1 and Fig. 2.

If we assume that $p_{L_e|A}(l|1) = p_{L_e|A}(-l|0)$ and that $p_{L_e|A}$ is Gaussian, then $I(A; L_e)$ becomes a function of $Var[L_e|A]$ only and (6) reduces to

$$\begin{aligned} I(A; L_e) &= 1 - E_{L_e|A=1}\left[\log_2\left(1 + e^{L_e}\right)\right] \\ &= 1 - E_{L_e|A=0}\left[\log_2\left(1 + e^{-L_e}\right)\right] \end{aligned} \tag{7}$$

IV. Convolutional Code Concatenated with APPM

When iteratively decoding a serially concatenated system, the extrinsic information from the inner SISO is fed to the outer SISO as a priori information. As can be envisioned from Fig. 1 and Fig. 2, for continuous improvement in the error rate with iterations, the output mutual information from the outer SISO has to be greater than the previous input mutual information to the inner SISO at each iteration. As proposed in [17], one can view the interaction between the two decoders by plotting $I(A; L_e)$ as a function of $I(A; L)$ for the inner SISO alongside $I(A; L)$ as a function of $I(A; L_e)$ for the outer SISO. In order to drive the error rate to zero, the curve of the inner SISO has to lie above the curve of the outer SISO.

The input mutual information for a convolutional code must go to one in order to drive the output mutual information to one (or the error rate to zero). Since a PPM SISO will never reach an output mutual information of one, its EXIT curve will always be lower at some points than the EXIT curve of any convolutional code used for the outer code. The serial

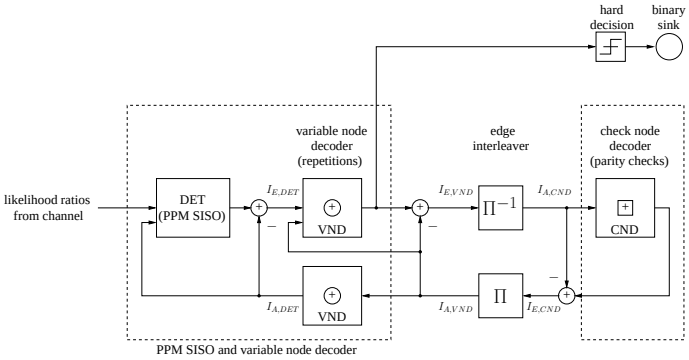

Fig. 3. Receiver model showing message passing flow and points for monitoring mutual information.

concatenation of a convolutional code with APPM instead of PPM, remedies this situation.

With the modulation fixed to be APPM, how should one choose the convolutional code? The optimizing criterion is to find a convolutional code whose EXIT curve will be lower than the EXIT curve of the APPM SISO at as low n_s as possible, while providing a concatenated code with sufficiently large minimum distance. The distance properties of the concatenated code depend on the interleaver design and block length (the EXIT charts do not predict distance properties), and are beyond the scope of this paper. In practice, for block lengths on the order of kilobits, the constraint length 3, rate $1/2$ $(5_o, 7_o)$ convolutional code was found to give the best results.

The resulting code design is the serial concatenation of the $(5_o, 7_o)$ convolutional code, followed by an interleaver, followed by APPM. We refer to this code as SCPPM. An EXIT analysis for this code predicts a threshold of $n_s = 1.94$ at $n_b = 0.2$.

V. LDPC Code Concatenated with PPM

Since an LDPC code may be designed to have an output mutual information of one for a input mutual information less than one, it lends itself to concatenation with either PPM or APPM. However, the APPM SISO requires a larger number of computations, and a much higher processing latency as will be discussed in Section VI-A. In order to limit the complexity, and, since we have seen no performance gain by using APPM in concatenation with an LDPC code, we choose to design an LDPC code to be used with PPM.

A. EXIT Analysis for LDPC Coded PPM

To determine the degree distribution of our LDPC code we follow the approach of [19]. Figure 3 shows a block diagram of the decoder illustrating the flow of messages (likelihoods) passed between an LDPC decoder and a PPM SISO and within the LDPC decoder. It also shows different points to monitor mutual information for our EXIT analysis. We refer the reader to [19] for further details on EXIT analysis for LDPC coded modulation. To facilitate our analysis, we will determine an EXIT curve for the PPM SISO (which will be denoted by DET to match the notation of [19]) combined with the variable nodes of the LDPC decoder, and another EXIT curve for the check nodes of the LDPC decoder. Hereinafter, we will denote the first curve by VND and the latter by CND. We will first determine the EXIT curves for a regular code with variable node degree d_v and check node degree d_c. We define the function $J(\cdot)$ as follows:

Definition 1: Let X be a random variable such that $Pr(X = 1) = Pr(X = -1) = \frac{1}{2}$, N be zero mean Gaussian noise with variance σ^2 and $Y = X+N$. We define the function $J\left(\frac{2}{\sigma}\right)$ to be the mutual information $I(X;Y)$. □

With the assumption that the LLR's exchanged between the PPM SISO and the variable nodes as well as those exchanged between the variable nodes and the check nodes are all Gaussian, then following the analysis and notation proposed in [19] and referring to Fig. 3 we can write

$$I_{A,DET} = J\left(\sqrt{d_v} J^{-1}\left(I_{A,VND}\right)\right)$$

$I_{E,DET}$ is computed from $I_{A,DET}$ by evaluating (6) numerically, obtaining the densities via simulation, i.e. without assuming that extrinsic LLR's of the PPM SISO are Gaussian. Following [19], $I_{E,VND}$ can then be given as:

$$I_{E,VND} = J\left(\sqrt{(d_v - 1)\left[J^{-1}\left(I_{A,VND}\right)\right]^2 + \left[J^{-1}\left(I_{E,DET}\right)\right]^2}\right)$$

For the check node curve, we use the following accurate approximation which is exact in the case of an erasure channel [19]:

$$I_{E,CND} \approx 1 - J\left(\sqrt{d_c - 1} \cdot J^{-1}\left(1 - I_{A,CND}\right)\right) \quad (8)$$

We may express (8) in terms of its inverse function to match the way it is plotted in Fig. 4:

$$I_{A,CND} \approx 1 - J\left(\frac{J^{-1}\left(1 - I_{E,CND}\right)}{\sqrt{d_c - 1}}\right)$$

Figure 4 shows our VND curves and CND curves for various values of d_v and d_c. These curves were generated using $M = 64$, $n_b = 0.2$ and $n_s = 1.995$. For a regular code to be converging at any particular n_s, its VND curve should be above its CND curve at this n_s. For an irregular code, the overall VND and CND curves are weighted averages of the different VND and CND curves depending on the degree distribution of the variable and check nodes. Fixing n_b, for any degree distribution, the n_s at which the overall VND curve touches the overall CND curve determines the threshold of the corresponding code. Minimization of this threshold is our optimization criteria.

B. LDPC Code Construction

An unconstrained optimization of the degree distributions of variable and check nodes may be unnecessarily complex, and could yield impractical degree distributions. Moreover, some heuristic constraints are helpful to reduce the error floor of the resulting code, which is not strictly predicted via EXIT curves alone. Using the EXIT curves, we optimized the degree distribution of our rate $1/2$ LDPC code subject to the following

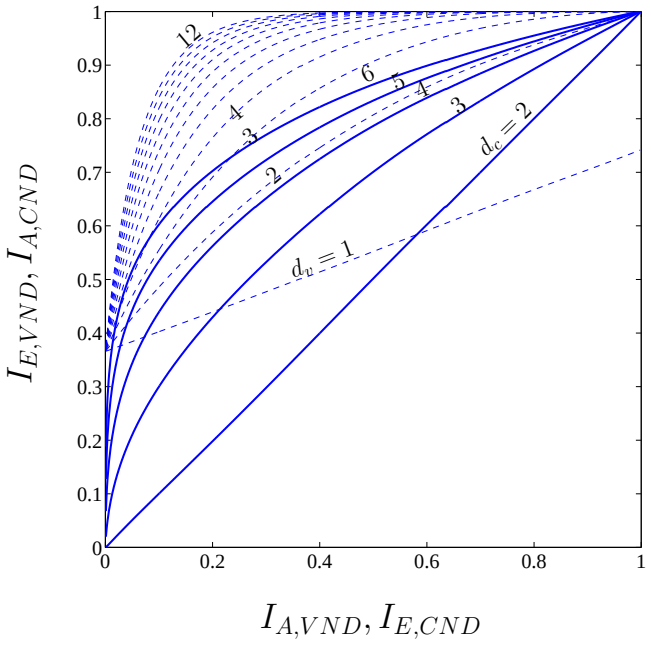

Fig. 4. VND and CND EXIT curves

heuristic constraints:

1) All check nodes have the same degree, to simplify our analysis.
2) The number of degree two variable nodes should be no more than 25% of the total number of variable nodes to avoid a high error floor.
3) The maximum variable node degree should be no more than 24. Increasing it beyond that increases the complexity with no significant performance gain.

Applying an exhaustive search for the best threshold subject to the above conditions, and rounding the degree distribution to rational numbers with a denominator less than 100, the following degree distribution was found: $21/88$ of the variable nodes are degree 2, $66/88$ are degree 3, $1/88$ are degree 24 and all check nodes are degree 6. Using this degree distribution, and fixing n_b to 0.2, the overall VND curve is above the overall CND curve for $n_s > 2.13$, which is the predicted threshold using this analysis technique.

C. Decoder Scheduling

For the SCPPM decoder, the outer code SISO is run once each time the APPM SISO is run, in order for each to have updated inputs. However, with LDPC-PPM, each decoding of the LDPC variable nodes sends updated messages to both the check nodes and the PPM SISO, and one has the option of scheduling these iterations in various manners. If the maximum allowed number of iterations between the PPM SISO and the LDPC decoder is small, scheduling does have some impact on performance. However, with the maximum number of iterations between the PPM SISO and the LDPC decoder sufficiently large (≈ 100), the number of iterations between the LDPC variable and check nodes each time the LDPC decoder is run (hereinafter denoted by LDPC iterations per PPM SISO iteration), has negligible impact on performance. However, it does have a large impact on decoding complexity. A detailed analysis of the complexity in the operation region of practical interest with M=64, shows that the number of both multiplications and additions are minimized by choosing two LDPC iterations per PPM SISO iteration.

VI. Comparison of the Two Coded Modulations

A. Latency

The decoder of SCPPM consists of two SISO's: one for the $(5, 7)$ convolutional code, and one for APPM. For the forward or backward paths of the BCJR algorithm [13], the codeword may be partitioned to smaller blocks and the blocks can be processed in parallel with a negligible loss in performance (up to some limit). However, within each block of the codeword, processing a symbol in the forward or backward paths depends on the results of processing the previous symbol. Therefore, within each block, the forward and backward paths have to be completed in a serial fashion, no matter how many parallel processors there are. This represents a major algorithmic computational difference between the APPM SISO and PPM SISO that is used in the proposed LDPC-PPM code. A PPM SISO can theoretically process all the symbols in parallel (although the number of processors may be too high for a practical implementation in today's technology), because there aren't any dependencies between symbols. Furthermore, the message passing algorithm of an LDPC decoder [20], as used in LDPC-PPM, lends itself naturally to parallel processing, as opposed to the SISO required for the $(5, 7)$ convolutional code required for SCPPM. Hence an LDPC-PPM decoder lends itself more naturally to parallel processing and a smaller decoding latency.

B. Operations Count

Table I gives the total number of operations per user bit for the complete SCPPM decoder and LDPC-PPM decoder for $M = 64$ and M=1024 at a word error rate of about 10^{-4}.

C. Performance

In this section we present simulation results for the SCPPM code and LDPC-PPM code described in this paper, as well as for an Accumulate-Repeat-Accumulate with repetition 4 (AR4A) LDPC code recently proposed for deep space RF communications [21]. The AR4A code is serially concatenated with PPM and iteratively decoded in the same manner as the LDPC-PPM code. All simulations are for the Poisson channel with $n_b = 0.2$ $photons/slot$ and $M = 64$. The codeword length is 8208 bits (i.e. 4104 user bits) for SCPPM, 8184 bits (i.e. 4092 user bits) for LDPC-PPM, and 8192 bits (i.e. 4096 user bits) for AR4A-PPM. Figure 5 shows the bit error rate P_b of these codes relative to the capacity of the Poisson PPM channel. At $P_b \approx 10^{-5}$ (corresponding roughly to a word error rate of 10^{-4} for the codes presented), SCPPM is about

TABLE I

TOTAL OPERATIONS COUNT PER USER BIT FOR SCPPM AND LDPC-PPM

	SCPPM		LDPC-PPM	
	Multiplications	Additions	Multiplications	Additions
Total operations per user bit for M=64	1419	2231	2042	2394
Total operations per user bit for M=1024	8352	22628	8024	19723

1 dB away from capacity, LDPC-PPM is about 1.4 dB away from capacity, while AR4A-PPM is about 2.2 dB away from capacity. The gap between SCPPM and LDPC-PPM agrees with the thresholds predicted for both codes in Sections IV and V-B.

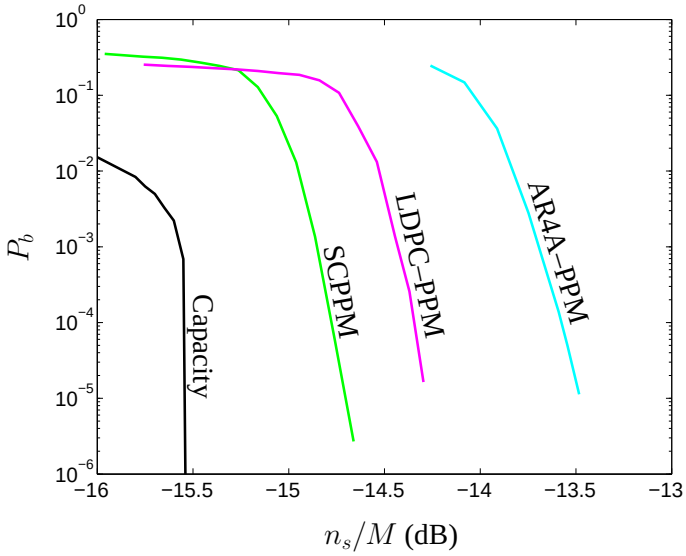

Fig. 5. Bit error rate performance of SCPPM, LDPC-PPM, and AR4A-PPM in comparison to capacity, $n_b = 0.2$, M=64

VII. CONCLUSION

We have developed a SCPPM code as well as an LDPC-based code for deep space optical communications. They operate at 1 dB and 1.4 dB away from capacity respectively. Either code would make a sensible solution depending on the PPM order, latency requirements, and the desired implementation architecture.

VIII. ACKNOWLEDGMENTS

Ken Andrews kindly provided us with an LDPC software encoder and decoder and other useful software tools.

The research described in this paper was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

REFERENCES

[1] A. D. Wyner, "Capacity and error exponent for the direct detection photon channel–Part II," *IEEE Transactions on Information Theory*, vol. 34, no. 6, pp. 1462–1471, Nov. 1988.

[2] S. Shamai, "Capacity of a pulse amplitude modulated direct detection photon channel," *IEE Proceedings I Communications, Speech and Vision*, vol. 137, no. 6, pp. 424–430, Dec. 1990.

[3] B. Moision and J. Hamkins, "Multipulse PPM on discrete memoryless channels," in *IPN Progress Report*, vol. 42-160, Feb. 2005.

[4] R. G. Lipes, "Pulse-position-modulation coding as near-optimum utilization of photon counting channel with bandwidth and power constraints," *DSN Progress Report*, vol. 42, no. 56, pp. 108–113, Apr. 1980.

[5] R. J. McEliece, "Practical codes for photon communication," *IEEE Transactions on Information Theory*, vol. IT-27, no. 4, pp. 393–398, July 1981.

[6] B. Moision and J. Hamkins, "Deep-space optical communications downlink budget: Modulation and coding," in *IPN Progress Report*, vol. 42-154, Aug. 2003.

[7] K. Kiasaleh, "Turbo-coded optical PPM communication systems," *Journal of Lightwave Technology*, vol. 16, no. 1, pp. 18–26, Jan. 1998.

[8] J. Hamkins, "Performance of binary turbo coded 256-PPM," *TMO Progess Report*, vol. 42, no. 138, pp. 1–15, Aug. 1999.

[9] M. Peleg and S. Shamai, "Efficient communication over the discrete-time memoryless Rayleigh fading channel with turbo coding/decoding," *European Transactions on Telecommunications*, vol. 11, no. 5, pp. 475–485, September-October 2000.

[10] D. Divsalar and F. Pollara, "Serial and hybrid concatenated codes with applications," in *Proc. Int. Symp. Turbo Codes & Related Topics*, Brest, France, Sept. 1997, pp. 80–87.

[11] B. Moision and J. Hamkins, "Coded modulation for the deep space optical channel: serially concatenated PPM," *IPN Progress Report*, vol. 42-161, 2005.

[12] A. Biswas and S. Piazzolla, "Deep-space optical communications downlink budget: Mars system parameters," *IPN Progress Report*, July 2003, to appear.

[13] L. Bahl, J. Cocke, F.Jelinek, and J.Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Transactions on Information Theory*, vol. 20, pp. 284–287, Mar. 1974.

[14] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, "A soft-input soft-output maximum a posteriori (MAP) module to decode parallel and serial concatenated codes," *Telecommunications and Data Acquisition Progress Report*, vol. 42-127, pp. 1–20, 1996.

[15] A. Ashikhmin, G. Kramer, and S. ten Brink, "Extrinsic information transfer functions: Model and erasure channel properties," *IEEE Transactions on Information Theory*, vol. 50, no. 11, pp. 2657–2673, Nov. 2004.

[16] T. J. Richardson, M. A. Shokrollahi, and R. L. Urbanke, "Design of capacity-approaching irregular low-density parity-check codes," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 619–637, Feb. 2001.

[17] S. ten Brink, "Convergence of iterative decoding," *Electronics Letters*, vol. 35, no. 10, pp. 806–808, May 1999.

[18] ——, "Convergence behaviour of iteratively decoded parallel concatenated codes," *IEEE Transactions on Communications*, vol. 49, no. 10, pp. 1727–1737, Oct 2001.

[19] S. ten Brink, G. Kramer, and A. Ashikhmin, "Design of low-density parity-check codes for modulation and detection," *IEEE Transactions on Communications*, vol. 52, no. 4, pp. 670–678, April 2004.

[20] R. G. Gallager, *Low-Density Parity-Check Codes*. Cambridge, MA: MIT Press, 1963.

[21] D. Divsalar, S. Dolinar, J. Thorpe, and C. Jones, "Constructing LDPC codes from simple loop-free encoding modules," in *Proceedings of IEEE International Conference on Communications*, vol. 1, Seoul, Korea, May 2005, pp. 658–662.